Journal of the Korean
OR/MS Society
Vol. 16, No.2, Dec., 1991

MAO모델을 사용한 관계 데이타베이스의 논리적 설계방법론

조동영* • 백두권** • 황종선**

A Logical Design Methodology for Relational Databases Using the MAO (Multiple Aspects-based Object) Model

Dong-Young Cho*, Doo-Kwon Baik,** Chong-Sun Hwang**

Abstract

In this paper, we present a stepwise design methodology for relational databases using a new conceptual data model, the MAO(Multiple Aspects-based Object) model.

Our methodology consists two steps: first, data requirements are conceptualized using the MAO model with concepts such as obiect types and aspect types; second, the MAO model is transformed into the third normal form in the relational model supported by commercial DBMSs. A top-down approach is used for the MAO modelling in the first step, and the transformation process in the second step can be automated. Our methodology supports easier and more accurate database design of real world than other methodologies using existing data models.

I. 서 론

관계모델은 그 표현구조의 간결성과 구현의 용이성에도 불구하고 모델링 개념의 단순성과 시맨틱 표현능력의 취약성때문에 실세계의 시맨틱 구조를 직접 관계모델로 모델링하는 데에는

* 고려대학교 전산과학과 박사과정 수료

** 고려대학교 전산과학과 교수

많은 어려움을 갖는다.

개념모델을 기반으로 하는 데이타베이스설계 방법들은 관계모델 대신 보다 다양하고 풍부한 시맨틱 표현능력을 지원하는 개념모델을 이용하여 실세계의 시맨틱 구조를 모델링하고, 이것을 DBMS(Data Base Management System)의 지원을 받는 관계모델로 변환한다[3, 6]. 일반적으로 개념모델들은 표현의 경제성, 무결성 유지의 용이성, 모델링의 융통성, 다양한 추상화 개념의 제공, 스키마의 그래프 표현등의 특성을 갖는다[1, 3].

그러나 ER(Entity Relationship) 모델이나 SDM(Semantic Database Model), SAM(Semantic Association Model)등 기존의 개념모델들은 실세계의 객체들에 대해 고정된 한 측면의 표현만을 허용하기 때문에 여러 측면에서 관측될 수 있는 객체를 모델링할 경우, 항상 데이타의 중복표현과 실세계와 모델 사이의 시맨틱-갭(semantic gap)를 내포하게 된다[7, 8, 9]. MAO(Multiple Aspects-based Object) 모델은 이러한 문제점을 해결하기 위한 다중측면 객체중심의 개념모델이다[8, 9].

따라서, 본 논문에서는 그림 1에서와 같이 실세계에 대한 보다 정확한 시맨틱 표현능력과 손쉬운 모델링 접근을 지원하는 MAO 모델을 이용하여 데이타베이스의 개념모델링을 수행하고, 이를 관계모델로 변환하는 2단계 접근전략에 의해 관계 데이타베이스를 설계한다. MAO 모델링은 하향식 접근방법을 이용함으로써 보다 손쉬운 모델링 수행을 도모하고, MAO모델에서 관계모델로의 변환과정은 자동화될 수 있음을 보인다.

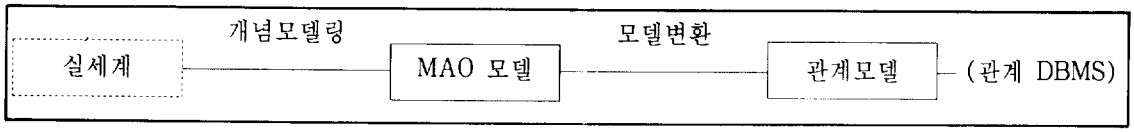


그림 1 MAO 모델을 이용한 관계 데이타베이스 설계방법

Ⅱ. MAO 모델의 개념

MAO 모델은 객체타입(object type)과 각 객체에 대한 측면(aspect) 개념을 이용하여 실세계의 객체들을 모델링한다. 객체타입은 모델구성자에 의해 독립적으로 인식될 수 있는 엔티티 타입(entity type)과 여러 객체타입들간의 관련성에 의해 추상적으로 인식되는 연관성 타입(association type)으로 구분된다. 엔티티 타입과 연관성 타입의 구별은 모델구성자의 뷰(view)에 의존한다. 예를들면, 대학교에 대한 모델링에서 '강의'라는 개념은 그림 2의 가)에서와 같이 하나의 독립된 엔티티 타입으로 표현될 수도 있고, 또 그림 2의 나)에서 처럼 '학생'및 '교수'라는 두 엔티티 타입의 관련성에 의한 연관성 타입으로 표현될 수도 있다. 그림 2의 가)에서 엔티티 타입 '강의'는 '학생'과 '교수'와는 독립적으로 존재하는 반면에 나)에서 연관성 타입 '강의'

는 '학생'및 '교수'에 종속해서 존재하게 된다.

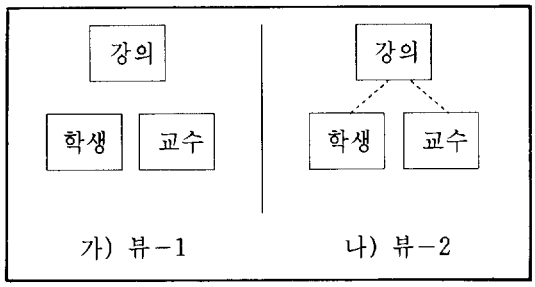
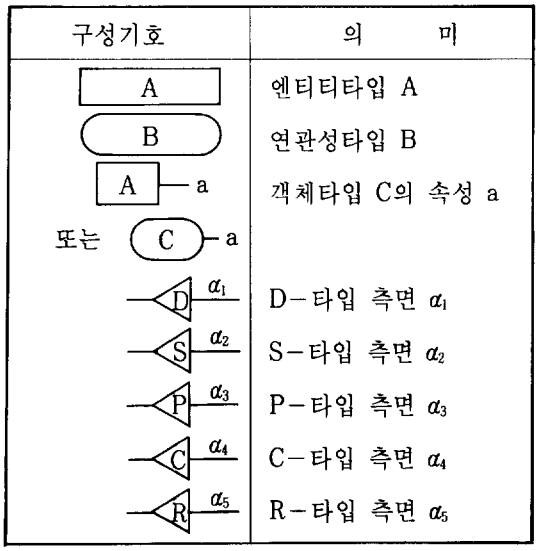


그림 2 엔티티타입과 연관성타입의 차이

측면은 엔티티 타입에 대한 한 이해관점을 나타내는 것으로, 주어진 엔티티 타입에 대한 한 측면의 결과로 또다른 여러 하위 엔티티 타입들이 인식된다. 한 엔티티 타입은 다수의 측면을 가질 수 있다. MAO 스키마는 MAO 도표로 표현되며, 그 주요 구성기호는 그림3과 같다.



(그림 3) MAO 도표의 주요 구성기호

측면은 엔티티 타입에 대한 뷰의 특성에 따라 D-타입, S-타입, P-타입, C-타입, R-타입으로 세분된다. D-타입 측면은 주어진 엔티티 타입이 여러 성분 엔티티타입들의 합성으로 이해될 수 있음을 나타낸다. S-타입 측면은 주어진 엔티티 타입의 객체들이 여러 중복가능한 엔티티 타입들로 관측될 수 있음을 나타내며, P-타입 측면은 주어진 엔티티 타입이 상호 배제적인 엔티티 타입들로 분할되어 이해될 수 있음을 나타낸다.

P-타입 측면은 하위 엔티티 타입들이 상호배제적이고, 하위 엔티티 타입들은 상위 엔티티 타입에 대해 완전해야 한다는 점에서 S-타입 측면과는 구별된다. C-타입 측면은 주어진 엔티티 타입이 상호 배제적인 엔티티 타입들의 합집합에 대한 부분집합으로 이해되고 있음을 나타낸다. R-타입 측면은 주어진 엔티티 타입이 다른 독립적인 엔티티 타입들과 연관성을 갖는다는 것을 의미한다. 그림 4는 ER모델의 관계성 표현이 MAO 모델에서는 R-타입 측면을 이용하여 표현될 수 있음을 보여준다.

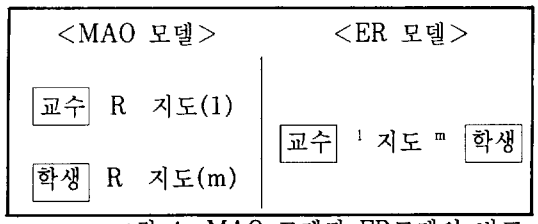


그림 4 MAO 모델과 ER모델의 비교 (관계성표현)

Ⅲ. MAO 모델의 하향식 설계방법: 개념적 MAO 스키마 설계

하향식 접근에 의한 MAO 모델의 설계과정은 그림 5와 같이 모두 다섯단계로 구성되며, 이를 자세히 설명하면 다음과 같다.

단계-1: 모델링의 목적에 부합되는 시스템 엔티티와 주요 엔티티 타입들을 선정하고, 각 엔티티 타입의 속성들과 키를 설계한다.

시스템 엔티티(system entity)는 MAO모델링의 출발점을 위해 도입한 특별한 엔티티 타입으로, 시스템 엔티티는 하나의 인스턴스만을 가지며, 하나 이상의 D-타입 측면만이 허용된다.

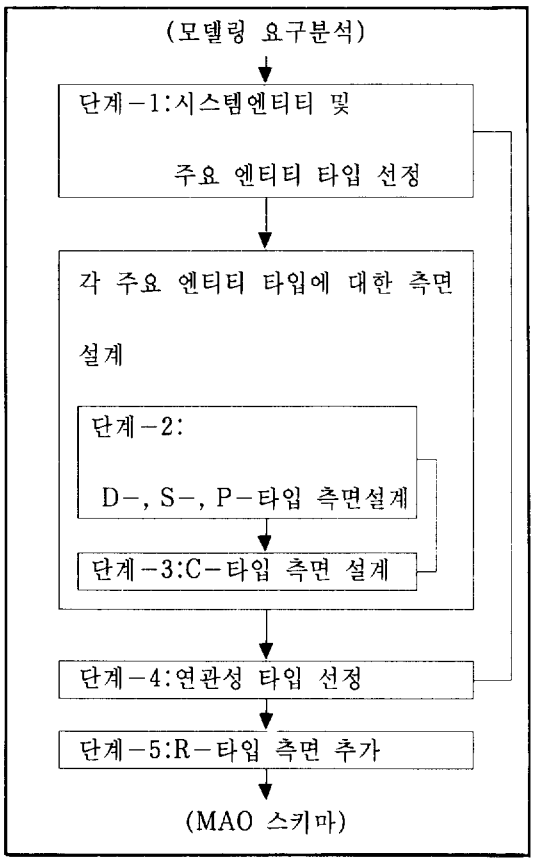


그림 5 MAO 모델의 하향식 설계방법

단계-1에서 선정되는 주요 엔티티 타입들은 그림6에서와 같이 시스템 엔티티의 '엔티티 구성' 측면으로 나타나고, 단계-4에서 설계되는 연관성 타입들도 이 시스템 엔티티의 D-타입 측면으로 나타낼 수 있다. 단계-1에서 선정되는 주요 엔티티 타입들은 상호배제적이어야 한다.

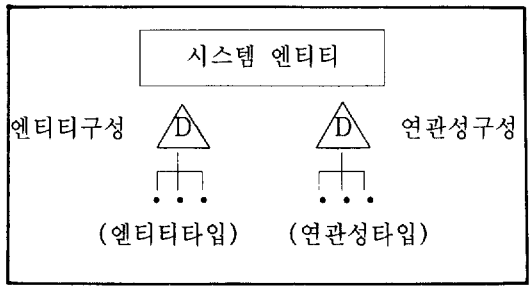


그림 6 시스템 엔티티의 두 D-타입 측면

단계-2: 단계-1에서 선정된 각 엔티티 타입에 대해, 필요하다면 D-타입과 S-타입, P-타입 측면을 설계한다.

D-타입과 P-타입, S-타입 측면의 구성 알고리즘은 알고리즘 3.1, 알고리즘 3.2와 같다. 단계-2에서, 각 측면의 설계를 보다 효율적으로 하기 위하여는 다음 사항들이 고려되어야 한다.

1) 엔티티 타입이 동일타입의 측면들을 다수 가질 경우, 각 측면에 의해 나타나는 관련 하위 엔티티 타입들의 집합은 측면에 따라 상호배제적이어야 한다. 즉, 엔티티타입 E가 α측면으로 관련 하위 엔티티 타입 집합 E_{11}, ……, E_{1n}이 선정되고, β측면으로 E_{21},……,E_{2m}이 선정되면 그리고 α측면과 β측면이 같은 D-타입 혹은 S-타입, P-타입이면, $\{E_{11}, \cdots, E_{1n}\}$과 $\{E_{21}, \cdots, E_{2m}\}$은 상호 배제적이어야 한다.

2) 한 엔티티 타입이 P-타입의 측면을 다수 갖는 경우, 이러한 측면들은 같은 P-타입의 단

일측면으로 통합이 가능하다. 따라서, 각 엔티티 타입은 P-타입 측면과 S-타입측면의 소유를 가능하면 하나로 하는 것이 바람직하다.

예 3.1: 그림 7에서 엔티티 타입 '사람'은 성별측면(P-타입)에서 '남자'와 '여자'로, 그리고 법률책임측면(P-타입)에서 '성인'과 '미성년자'로 모델링된다. 이 두 P-타입 측면은 각 측면을 순차적으로 적용함으로써 그림 7에서 보는 바와 같이 하나의 측면으로 통합될 수 있다.

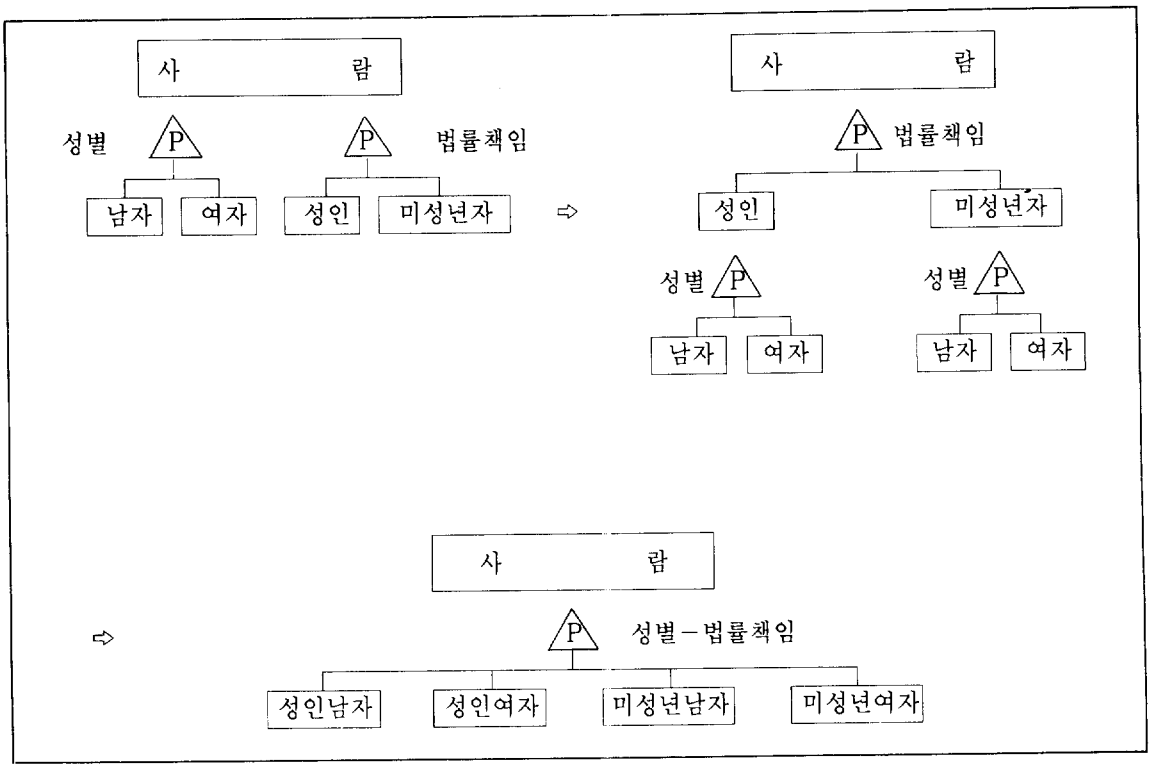


그림 7 다수의 P-타입 측면들은 P-타입의 단일측면으로 통합가능하다.

단계-3: 필요하다면, 단계-2까지에서 선정된 엔티티 타입들로부터 C-타입 측면을 갖는 새로운 엔티티 타입을 생성한다.

C-타입 측면의 구성 알고리즘은 알고리즘 3.3과 같다. C-타입 측면의 설계는 다음을 만족해야 한다.

1) C-타입 측면을 갖는 새로운 엔티티 타입의 생성에 참여하는 기존의 엔티티 타입들은 서로 배제적이어야 한다.

알고리즘 3.1 Decomposition_Aspect(E, α)

{엔티티 타입 E의 D-타입 측면 'α'의 구성 알고리즘, E, E_i: 엔티티 타입,
OWNER(α): 측면 α의 소유 엔티티 타입,
REL(α): 측면 α에 의해 관련되는 엔티티 타입 집합}

1. OWNER(α)←E;
2. REL(α)← ϕ ;
3. i←1;
4. Selected←. TRUE.;
5. 엔티티 타입 E_i를 선정한다.;
6. FOR j = 1 TO i−1
   DO IF($E_i \cap E_j \neq \phi$) THEN BEGIN
   Selected←.FALSE.;
   GO TO 9
   END;
7. REL(α)←REL(α) ∪ {E_i};
8. E_i의 속성을 설계한다.;
9. IF .NOT. (엔티티타입 추가 필요성) THEN GO TO 12;
10. IF Selected=.TRUE. THEN i←i+i;
11. GO TO 5;
12. STOP {D-타입 측면 α의 구성 종료}

알고리즘 3.2 Specialization_Aspect(E, β)

{엔티티 타입 E의 특수화타입 측면 'β'의 구성 알고리즘,
E, E_i: 엔티티 타입,
OWNER(β): 측면 β의 소유 엔티티 타입
REL(β): 측면 β에 의해 관련되는 엔티티 타입 집합}

1. OWNER(β)←E;

2. REL(β)← ϕ ;

3. 엔티티 타입 E_i를 선정한다.;

4. IF Domain(E_i)⊆Domain(E) THEN GO TO 8;

5. REL(β)←REL(β)∪{E_i};

6. E_i의 속성을 설계한다.;

7. IF (E의 측면설계 필요성)

THEN call decomposition_aspect(E_i, β_i)

혹은 specialization_aspect(E_i, β_i);

8. IF (엔티티타입 추가선정 필요성) THEN GO TO 3;

{특수화 타입의 결정}

9. IF ($\forall$i, j, Domain(E_i)∩Domain(E_j) = ϕ

and ∪Domain(E_i)∩Domain(E_j) and E_i∈REL(β))

THEN TYPE(β)←'P'

ELSE TYPE(β)←'S';

10. STOP {특수화-타입 측면 β의 구성 종료}

2) C-타입 측면을 갖는 새로운 엔티티 타입을 생성할 때, 엔티티타입 노드와 측면타입노드에 의한 동일계층트리의 경로상에 나타나는 두 엔티티 타입은 함께 새로운 엔티티 타입의 C-타입 측면에 의한 관련 엔티티 타입으로 참여할 수 없다.

3) C-타입 측면을 갖는 엔티티 타입의 인스턴스들은 관련된 상위의 엔티티 타입들중 반드시 한 타입의 인스턴스이어야 한다. 따라서 C-타입 측면을 갖는 엔티티 타입은 각 인스턴스가 소속되는 상위 엔티티 타입을 나타내는 범주속성(category attribute)을 포함해야 한다.

예 3.2:그림 8에서 엔티티 타입 '연구원'은 '신

분'측면(C-타입)으로 '교수', '대학원생'과 관련되어, 범주속성 'category'를 가질 수 있다. 즉, '연구원'의 한 인스턴스 x는 category(x)='대학원생'이면, x는 '대학원생'으로부터 모든 속성과 측면구조를 상속받는다.

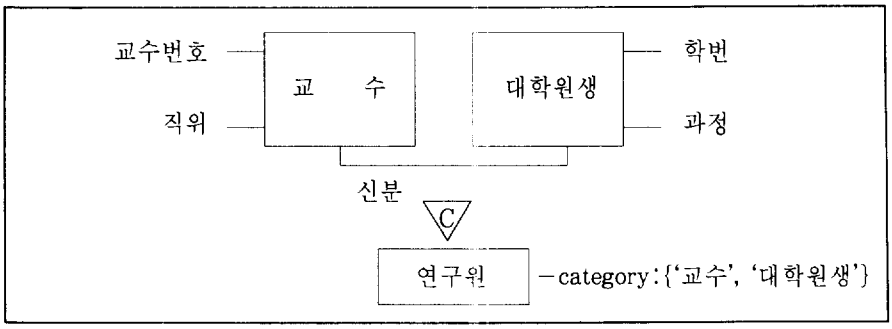


그림 8 C-타입 측면과 범주속성

알고리즘 3.3 category_aspect(E, γ, E_1,…,E_n)

{엔티티 타입 E_1,……, En 으로부터 C-타입 측면 γ를 갖는 새로운 엔티티 타입 E의 구조를 설계한다.

E: 단계-2까지 선정된 엔티티 타입들의 모임}

{E,E_1,…E_n의 타당성 검사}

```
1. IF E∈ E THEN BEGIN
                    WRITE 'C-타입 구성 오류';
                    GO TO 7
                  END;
2. FOR i=1 TO n DO
   BEGIN
       IF Ei∉E THEN BEGIN
                      WRITE 'C-타입 구성 오류';
                      GO TO 7
                    END;
       FOR j=1 TO n DO
```

IF($i \neq j$ and $E_i \in REL(\gamma)$ for $\forall \quad \gamma \in aspect(E_j)$)

THEN BEGIN

WRITE 'C-타입 구성 오류';

GO TO 7

END

END;

3. OWNER(γ)←E;

4. TYPE(γ)←'C';

5. E의 범부속성을 포함하는 속성집합을 설계한다;

6. IF (E의 측면설계 필요성)

THEN call algorithm 3.1 혹은 algorithm 3.2;

7. STOP {C-타입 측면 γ의 구성 종료}

단계-4: 단계-3까지에서 선정된 엔티티 타입들 사이의 관계성을 토대로 연관성 타입들을 선정한다.

동일 계층구조 경로상에 나타나는 두 엔티티 타입이 동시에 참가하는 연관성 타입은 정의될 수 없다. 연관성 타입의 설계시 고려해야 할 사항은 다음과 같다.

1) 연관성 타입에 참가하는 각 엔티티 타입이 반드시 상호 배제적일 필요는 없다. 그리고 동일 엔티티 타입이 한 연관성 타입에 한번 이상 참여할 수도 있다.

2) 연관성 타입의 속성집합은 각 참가자의 키를 포함해야 한다.

3) 연관성 타입의 각 참가자는 자신의 참가 카디날리티를 명세해야 한다.

4) 단지 두 참가자가 각각 1의 참가 카디날리티를 갖고, 또 그 연관성 타입이 참가자의 키 이외에 별도의 속성집합을 갖지 않을 경우, 이러한 연관성 타입은 제거될 수 있다.

5) 연관성 타입은 참가자들과는 독립적인 속성집합을 가질 수 있다.

6) 연관성 타입에 참가하는 참가자들의 수는 제한되지 않는다.

단계-5: 단계-4에서 선정된 연관성 타입들을 토대로, 단계-3까지의 각 엔티티 타입에 R-타입 측면을 추가한다.

연관성 타입의 각 참가자는 그 연관성 타입을 암시하는 R-타입 측면을 가져야 한다. 따라서, 단계-5에서는 단계-4에서 선정된 연관성 타입들을 모두 고려함으로써 각 엔티티 타입의 모든 R-타입 측면들을 설계할 수 있다.

예 3.3 : 그림9는 한 대학의 학사관리업무에 대한 MAO모델링 예를 나타낸다. 그림 9에서 '대학'은 시스템엔티티이고 엔티티 타입 '사람', '교과목', '학과'는 단계-1에서 선정된 주요 엔티티 타입들이다. 단계-2에서 주요 엔티티타입 '사람'은 신분측면(P-타입)으로 '교수'와 '학생'으로 분할되고 '학생'은 다시 고급과정측면(S-타입)으로 '대학원생'으로 특수화되었다. 그리고 다음 단계에서 연관성 타입 '논문지도', '수업', '강의개설'이 설계되고 이들의 각 참가자에는 R-타입 측면이 추가되었다. 그림 9에서 C-타입 측면은 고려되지 않았다.

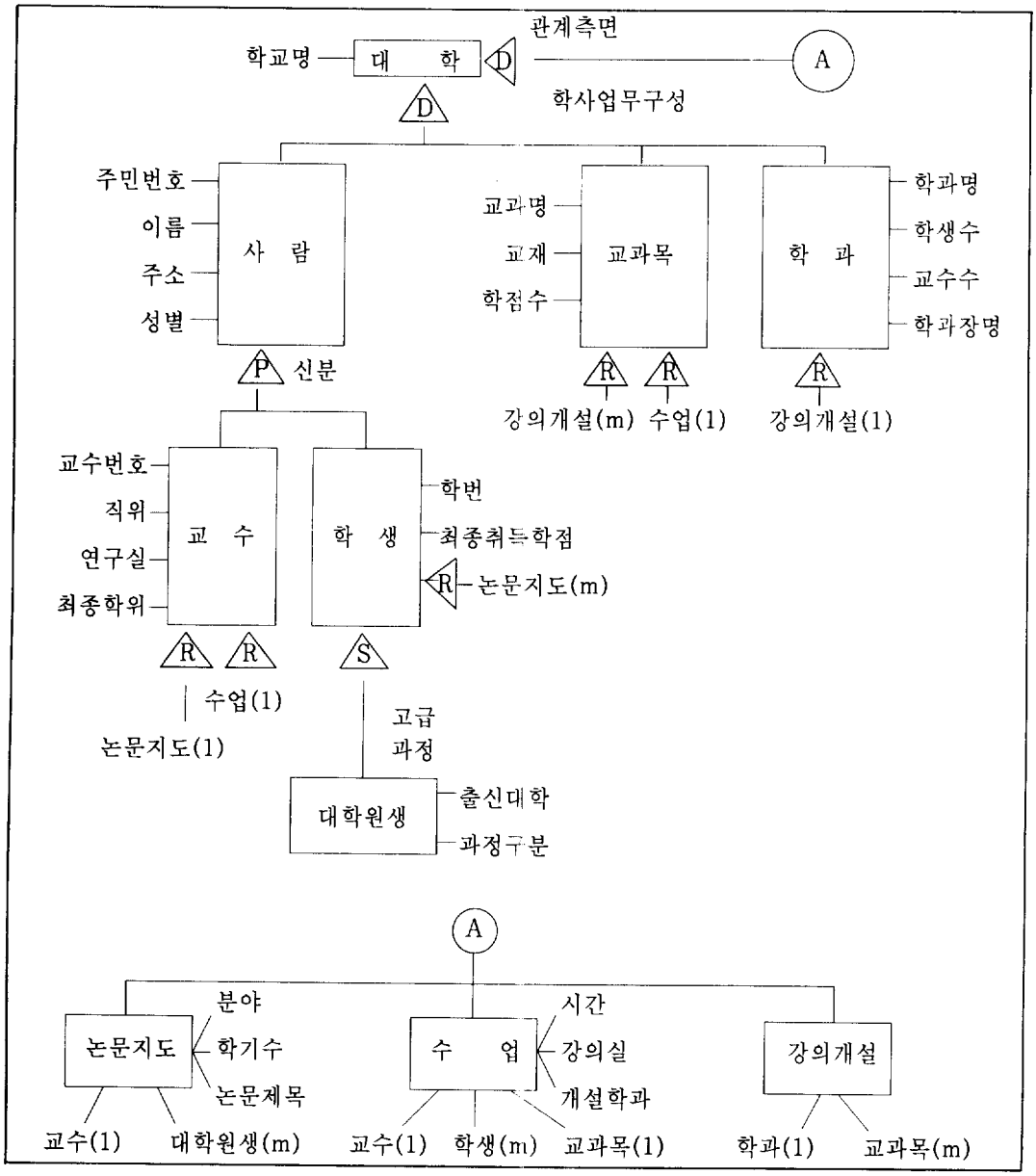


그림 9 대학학사업무에 대한 MAO 모델링 예

Ⅳ. MAO모델의 관계모델 변화: 논리적 관계데이타베이스 설계

4.1 MAO모델의 데이타 종속성 분석

MAO 스키마를 관계모델에 의한 논리스키마로 변환하기 위해서는 먼저 MAO 스키마에 나타나는 데이타 종속성의 분석이 필요하다. MAO 모델의 데이타종속성에는 객체타입 수준의 데이타종속성으로 EFD(Entity Functional Dependency) 관계와 AFD(Attribute Functional Dependency) 관계가 있으며, 인스턴스 수준의 데이타 종속성으로 존재종속성(ED: Existence Dependency) 관계가 있다.

정의 4.1 : 두 엔티티 타입 E_1, E_2에 대하여, 이들 사이에 연관성 타입 A가 존재하고 , A를 통해 E_1의 한 인스턴스 객체가 E_2의 한 인스턴스 객체에만 관련될 때, 이 두 엔티티 타입 E_1, E_2 사이에는 '엔티티 함수적 종속성(EFD) 관계가 존재한다'라고 하고 EFD:$E_1 \rightarrow E_2$로 표시한다.

정의 4.2: 두 속성 A, B에 대하여, 한 객체의 속성 A의 값이 이 객체의 속성 B의 값을 유일하게 결정할 때, 두 속성 A, B 사이에는 '속성 함수적 종속성(AFD) 관계가 존재한다'라고 하고, AFD: A→B로 표시한다.

정의 4.3 두 객체타입 O_1, O_2에 대하여, O_2의 한 인스턴스 객체가 존재하기 위해서는 먼저 O_1에 그에 관련되는 인스턴스 객체가 반드시 존재해야 할 때, 두 객체타입 O_1, O_2 사이에는 '존재종속성(ED) 관계가 존재한다'라고 하고, ED: $O_1 \rightarrow O_2$로 표시한다. 객체타입 O_1, O_2, O_3에 대해, ED: $O_1 \rightarrow O_2$, ED: $O_1 \rightarrow O_3$일 때, 이를 간단히 ED: $O_1 \rightarrow \{O_2, O_3\}$로 표시한다.

MAO모델에서 객체타입들 사이의 존재종속성은 다음의 다섯가지 경우에 나타난다.

1) 엔티티 타입 E가 D-타입 측면 α_1를 가질 경우, ED: $E \rightarrow REL(\alpha_1)$[3] 관계가 존재한다.

2) 엔티티 타입 E가 S-타입 측면 α_2를 가질 경우, ED: $E \rightarrow REL(\alpha_2)$ 관계가 존재한다.

3) 엔티티 타입 E가 P-타입 측면 α_3를 가질 경우, ED: $E \rightarrow REL(\alpha_3)$ 관계가 존재한다. 그리고 ED: $\cup E_i \rightarrow E(E_i \in REL(\alpha_3))$ 관계도 존재한다.

4) 엔티티 타입 E가 C-타입 측면 α_4를 가질 경우, ED: $\cup E_i \rightarrow E(E_i \in REL(\alpha_4))$ 관계가 존재한다.

5) 엔티티 타입 E가 R-타입 측면 α_5를 가질 경우, ED: $E \rightarrow REL(\alpha_5)$ 관계가 존재한다. 여기서 $REL(\alpha_5)$는 E가 참가하는 연관성 타입이 된다.

예 4.1 : 그림 9에서 엔티티 타입 '학생'은 고급과정(S-타입)측면과 논문지도(R-타입) 측면을 갖는다. 이것은 위의 2)와 5)에 의해 다음의 존재종속성 관계가 존재함을 의미한다.

3) 엔티티 타입 E의 측면 α_i에 의해 나타나는 객체타입들의 집합.

ED: 학생→대학원생
('고급과정' 측면으로부터 도출)
ED: 학생→논문지도
('논문지도'측면으로부터 도출)

데이타 모델이 실세계의 시맨틱을 명확하게 표현하기 위해서는 모델내에서의 모든 데이타 종속성들이 명시적으로 표현될 수 있어야 한다.

정의 4.4: MAO 스키마에 존재하는 엔티티 타입 E에 대하여, E의 키가 존재하고, 키를 제외한 나머지 속성들 사이에 어떠한 AFD 관계도 존재하지 않을 때, 엔티티 타입 E를 'well-defined entity type'이라고 한다.

정의 4.5: MAO 스키마에 존재하는 연관성 타입 A에 대하여 A의 모든 속성들 사이에 어떠한 AFD 관계도 존재하지 않을 때, 연관성 타입 A를 'well-defined association type'이라고 한다.

MAO 스키마를 구성하는 모든 엔티티 타입 및 연관성 타입들이 well-defined 되었을 때, 이러한 MAO 스키마를 'well-defined MAO schema'라고 한다. well-defined MAO 스키마에서 모든 데이타 종속성들은 명시적으로 표현된다. 그림 9는 well-defined MAO 스키마이다.

4.2 관계모델로의 변환방법

개념적 데이타베이스 스키마로 설계된 MAO 스키마는 DBMS의 지원을 받는 관계모델 스키마로 변환될 수 있다. MAO 스키마는 관계모델에서 메타관계 테이블(metarelation table)과 데이타 관계스킴(data relation scheme)으로 표현된다. 메타관계 테이블은 관계모델이 MAO모델의 구조정보를 유지하기 위하여 필요한 것이며, 데이타 관계스킴은 인스턴스 정보를 유지하기 위하여 필요한 것이다. 메타관계 테이블은 MAO스키마를 관계스키마로 변환할 때 생성된다. 그리고 데이타 관계스킴들 사이의 제약사항(interrelation schema constraints)은 4.1절에서 설명한 관계스킴들간의 존재종속성 관계로 표현된다. MAO 스키마를 관계스키마로 변환시키는 방법은 모두 다섯단계로 구성된다. 처음 세 단계는 관계스킴[4]을 구성하는 단계이고 나머지 두 단계는 구성된 관계스킴들 사이의 제약조건들을 생성하는 단계이다.

단계-1: MAO모델의 엔티티 타입, 연관성 타입, 측면정보를 유지하는 메타테이블을 위한 관계스킴을 구성한다.

각 메타관계 테이블 스킴들은 3 NF을 만족하고 속성의 널값을 불허하기 위하여 세분된다. 이를 위한 각 관계스킴의 구조는 다음과 같다.

ENTITY_TABLE1=({E_NAME, KEY}, {E_NAME})
ENTITY_TABLE2=({E_NAME, ATTRIBUTE}, {E_NAME})
ENTITY_TABLE3=({E_NAME, UPPER_ENTITY}, {E_NAME})
ENTITY_TABLE4=({E_NAME, WITH_ASPECT}, {E_NAME})

ASSOCIATION_TABLE1=({A_NAME, ATTRIBUTE}, {A_NAME})

ASSOCIATION_TABLE2=({A_NAME, PARTICIPANT}, {A_NAME})

ASPECT_TABLE1=({ASP_NAME, TYPE}, {ASP_NAME})

ASPECT_TABLE2=({ASP_NAME, RELATED_ENTITY}, {ASP_NAME})

단계-2: 각 엔티티 타입 E에 대해, 대응되는 관계스킴 R를 구성한다. 엔티티 타입 E에 대해, ENTITY_TABLE1, 2, 3, 4 각각의 튜플을 생성하고, E에 대응하여 관계스킴 R을 구성한다. E의 속성집합은 대응하는 관계스킴 R로 직접 사상되며, E의 키는 R의 키가 된다. 만약 E에 키가 존재하지 않는다면(이 경우, E는 반드시 상위엔티티 타입의 'P'혹은 'S'-타입 측면으로 설계되어야 한다), E는 상위엔티티 타입의 키를 상속받게 된다. 그리고 E가 소유하는 각 측면에 대해, ASPECT_TABLE1, 2 각각의 튜플을 생성한다.

예 4.2 : 그림 9에서 엔티티타입 '대학원생'은 키를 갖지 않는다. 따라서 대응되는 관계스킴은 상위 엔티티 타입인 '학생'의 키를 상속받아 다음과 같이 구성된다.

GRADUATE_REL=({학번, 출신대학, 과정구분}, {학번})

단계-3: 각 연관성 타입 A에 대해, 대응하는 관계스킴 R을 구성하고, A에 관련된 존재종속성(ED) 관계를 도출한다.

연관성 타입 A에 대해, ASSOCIATION_TABLE1, 2 각각의 튜플을 생성하고, A에 대응하여 관계스킴 R을 구성한다. A의 속성집합은 대응되는 R로 직접 사상되고 A의 각 참가자들의 키가 R의 속성집합에 추가되며, 이러한 참가자들의 키의 합집합이 R의 키가 된다. 그리고 R과 A의 각 참가자에 대응되는 관계스킴들 사이의 존재종속성 관계가 생성된다.

예 4.3 : 그림 9에서 엔티티타입 '교수', '학생', '교과목'은 연관성타입 '수업'에 참가한다. 따라서, 연관성 타입 '수업'에 대응하는 관계스킴 CLASS_REL의 속성집합에는 각각 '교수', '학생', '교과목'에 대응하는 관계스킴들의 키가 추가되고 이들의 조합이 CLASS_REL의 키가 된다. '수업'에 의해 구성되는 관계스킴의 구조와 존재종속성은 다음과 같다.

CLASS_REL=({교수번호, 학번, 교과명, 시간, 강의실, 개설학과}, {교수번호, 학번, 교과명})

ED: {교수, 학생, 교과목}→수업

단계-4:EFD관계를 이용하여, 단계-3에서 구성된 관계스킴들의 키를 수정한다.

이 단계에서는 각 연관성 타입에 참가하는 엔

4) 관계스킴의 구조는 R=(A, K) 형태로 나타낸다. 여기서, R은 관계스킴이름, A는 R의 속성집합, K는 R의 키 집합이다.

티티 타입들의 참가 카디날리티를 토대로, EFD 관계를 이용하여 연관성 타입에 대응하는 관계스킴의 키를 수정한다. 각 관계스킴의 키의 수정방법은 다음과 같다.

1) 연관성 타입 A에 참가하는 엔티티 타입들 사이에 단지 하나의 EFD관계가 존재할 경우에는 '1'이 참가카디날리티를 갖는 엔티티 타입에 대응하는 관계스킴의 키를 A에 대응하는 관계스킴의 키에서 제거한다.

2) 한 연관성 타입 A에 참가하는 엔티티 타입들 사이에 둘 이상의 EFD관계가 존재할 경우에는 위의 1)의 경우를 반복적으로 적용한다. 매번 1)의 경우를 적용할 때마다 후보키가 구성되고, 이들중 하나가 주요 키로 선택된다.

예 4.4 : 그림 9에서 연관성 타입 '수업'과 관련해서 다음의 EFD관계가 도출된다.

EFD: 학생→교수, EFD: 교과목→교수

따라서, 단계-3에서 구성된 관계스킴 CLASS_REL의 키는 위의 각 EFD관계로 얻어지는 후보키 {학번, 교과목명}, {학번, 교수번호} 중 어느 하나로 수정될 수 있다.

CLASS_REL=({교수번호, 학번, 교과명, 시간, 강의실, 개설학과}, {학번, 교과명})

단계-5: 각 측면구조로부터, 관계스킴들간의 관계성을 도출한다.

단계-3에서 생성된 존재종속성들은 R-타입 측면구조에 의한 존재종속성들이다. 이 단계에서는 4.1절에서 설명한 R-타입 이외의 각 측면구조로부터 생성될 수 있는 존재종속성 관계들이 도출된다. 즉, ASPECT_TABLE1, 2의 각 튜플로부터, 관계스킴들간의 존재종속성 관계를 도출한다.

예 4.5 : 그림 9의 학사업무구성측면(D-타입), 신분측면(P-타입), 고급과정측면(S-타입)으로부터 도출되는 존재종속성 관계는 다음과 같다.

ED: 대학→{사람, 교과목, 학과}

ED: 사람→{교수, 신분}

ED: 교수∪신분→사람

ED: 학생→대학원생

4.3 자동스키마 변환시스템의 구현

4.2절에서 설명한 스키마 변환방법을 지원하는 자동스키마변환시스템은 Turbo C 2.0으로 구현되었다. 이 시스템은 well-defined MAO 스키마를 입력으로 받아서 이에 대응하는 3NF의 관계스킴과 이러한 관계스킴들간의 존재종속성 관계 그리고 MAO스키마의 구조정보를 위한 메타테이블들을 출력한다. 이 시스템의 구성은 그림 10과 같다.

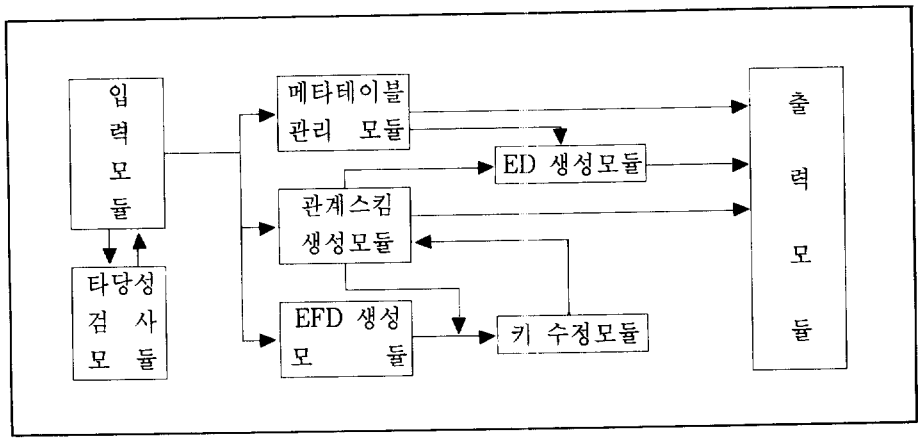


그림 10 MAO모델에서 관계모델로의 스키마 자동변환시스템의 구성

V. 결 론

본 논문에서는 다중측면 객체중심의 데이타베이스 모델링을 지원하는 새로운 데이타 모델인 MAO모델을 토대로 데이타베이스 개념스키마를 설계하고 이를 관계모델로 변환하는 단계적 관계 데이타베이스 설계방법론을 제시하였으며, 특히 MAO모델에서 관계모델로의 변환은 자동화될 수 있음을 보였다.

MAO모델은 기존의 다른 모델들과는 달리 단일객체에 대해 다수의 측면표현을 지원함으로써 실세계의 시맨틱에 대한 보다 정확한 모델표현을 가능하게 하고, 측면개념을 통해 객체타입들간의 상속성을 허용함으로써 데이타의 중복표현을 최소화할 수 있게 하였다. 따라서 MAO모델을 기반으로 하는 본 설계방법은 효율적인 관계 데이타베이스의 구축을 보다 용이하게 할 것이다.

참 고 문 헌

[1] Dionysios C. Tsichritzis and Frederick H. Lochovsky, *Data Models*, Prentice-Hall, Inc., 1982.

[2] J. Paredens, P. de Bra, M. Gyssens, and D. Van Gucht. *The Structure of the Relational Database Model*, Springer-Verag Berlin Heidelberg 1989.

[3] Joan Peckham and Maryanski, "Semantic Data Models", *ACM Computing Surveys*, Vol. 20, No. 3, September 1988.

[4] Michael R. Blaha, William J. Premerlani and James E. Rumbaugh, "Relational Database Design Using an Object-oriented Methodology", *Communications of the ACM*, Vol. 31, No. 4, April 1988.

[5] Myung-Joon Kim, Won-Ung Lee and Jean-Claude Derniame, "Automatic Relational Data Base Designs

By Transformation of the Entity_Relationship Model", the Second Inter. Conference on Computers and Applications, Beijing, China, Jun. 1987.

[6] Toby J. Teorey, Dongqing Yang and James P. Fry, "A Logical Design Methodology for Relational Databases Using the Extended Entity-Relationship Model", *Computing Surveys*, Vol. 18, No. 2, June 1986.

[7] 김창화, "EA 모델에 의한 지식표현 모델링", 고려대학교, 박사학위 논문, 1989.

[8] 조동영, 황종선, "다측면 객체모델 이론: 개념적 데이타 모델링에 대한 수학적 접근",「한국정보과학회, 91 봄 학술발표논문발표집,」Vol. 18, No. 1, 1991.

[9] 조동영, 손진곤, 백두권, 황종선, "다측면 객체모델의 정형화에 관한 연구",「한국 경영과학회 컴퓨터 활용 연구회, 춘계학술발표논문집」, 1991.